Department of Computer Science
George Mason University Technical Reports

4400 University Drive MS#4A5
Fairfax, VA 22030-4444 USA
http://cs.gmu.edu/ 703-993-1530

Discerning Machine Learning Degradation via Ensemble Classifier Mutual Agreement Analysis

Charles Smutz
csmutz@gmu.edu

Angelos Stavrou
astravrou@gmu.edu

Technical Report GMU-CS-TR-2015-11

Abstract

Machine learning classifiers are a crucial component of modern malware and intrusion detection systems. However, past studies have shown that classifier-based detection systems are susceptible to evasion attacks in practice. Improving the evasion resistance of learning based systems is an open problem.

In this paper, we analyze the effects of mimicry attacks on real-world classifiers. To counter such attacks, we introduce a novel approach that not only exposes attempts to evade the classifier, but also evaluates the ability of the system to operate reliably. We advance mutual agreement analysis for ensemble classifiers: during the detection operation, when a sufficient number of votes from individual classifiers disagree, the ensemble classifier prediction is shown to be unreliable. This method allows the detection of classifier evasion without additional external information.

To evaluate our approach, we used thousands of both malicious and benign PDF documents. Applying our method to two recent evasion attack studies, we show that most evasion scenarios are detected. Furthermore, we show that our approach can be generalized to weaken the effectiveness of the Gradient Descent and Kernel Density Estimation attacks against Support Vector Machines (SVM) by creating an ensemble classifier using many base SVMs. We discovered that feature bagging is the most important property for enabling mutual agreement based evasion detection.

1 Introduction

Machine learning techniques have emerged as one of the primary detection techniques against a wide-range of malfeasance and malicious activities in general including intrusion detection systems [36, 40, 6], clustering of malware families [10, 24], detection of malicious downloads [15, 38], detection of account misuse in social networks [51, 18], and detection of commonly exploited file formats such as Java archives [42] and documents [29, 44, 28]. Moreover, statistical or machine learning techniques have been used successfully for years to identify SPAM [41, 14, 25].

One of the main weaknesses of systems that use machine learning classification in adversarial environments is their susceptibility to evasion attacks. With evasion attacks, we refer to the classes of attacks that take advantage of the knowledge of how the machine learning system operates, and in many cases utilize access to the training set and features to evade detection passively or actively [19, 37, 52, 12, 11].

Mimicry attacks thwart detection by making the attack data appear benign according to the model used by the intrusion detection system. Often this is achieved by hiding overtly malicious content through encoding or encryption [48, 32] or minimizing the footprint of malicious content through data misuse or code re-use attacks [43, 21]. For instance, content aligning with a benign observation is added to cover up or drown out the malicious content. Often many detection systems are evaded by exploiting differences in the detection system and the systems being protected [20, 23]. Even if operational details of defense systems are kept secret, enough knowledge to conduct evasion can often be obtained solely from external testing [22]. With all of these potential evasion vectors, preventing detection evasion remains an open problem.

Our approach is not to prevent all possible evasion attacks, but to introduce a mechanism that provides detection of poor classifier performance. We analyze the use of introspection in an ensemble classifier to detect when the classifier provides unreliable results at classification time. The use of ensemble classifier mutual agreement analysis relies on the intuition that when individual classifiers in an ensemble vote for the same prediction, the prediction is likely to be accurate. When sufficient number of the votes are in opposition, then the classifier prediction is not trustworthy. In this state of internal classifier disagreement, the detector returns the outcome of “uncertain” instead of a prediction of benign or malicious. In operation, the classification rate of the detector is improved at the cost of a small portion of the samples being labeled as uncertain, indicating that the classifier is not fit to provide an accurate response. This separation of accurate predictions from uncertain predictions is possible because the majority of the misclassifications, including evasion attempts, have a classifier voting score distribution distinct from the accurate predictions. Our study indicates that feature bagging in an ensemble classifier, or constructing many individual classifiers with randomized subsets of the whole feature set, is critical to providing this discriminatory power.

To evaluate our technique, we study evasion attacks against PDFrate. PDFrate uses features derived from document structure and metadata fed into a Random Forest classifier to detect Trojan PDFs. PDFrate is used in real world intrusion detection systems and can be evaluated by the public through submissions to pdfrate.com. There are many choices for malware detectors on which to demonstrate mutual agreement analysis. PDFrate was selected because it is publicly accessible, well documented, uses an ensemble classifier which returns the raw voting score, and has been subjected to multiple recently published mimicry attacks [30, 39].

Our evaluation includes application of mutual agreement analysis to over 10,000 documents sourced from VirusTotal [7] and hundreds of malicious documents in nine unique evasion scenarios from two independent evasion studies. Lastly, we seek to demonstrate that we are able to defeat the Gradient Descent and Kernel Density Estimation (GD-KDE) attack, which is highly successful against a traditional Support Vector Machine (SVM) classifier.

Our contributions are:

- A simple but effective method of detecting classifier degradation in practice without resort to external ground truth data
- Evaluation against two recent evasion attack studies
- Generalization of our approach by defeating the GD-KDE attack against SVM classifiers using an ensemble SVM classifier created using feature bagging

2 Related Work

Adversarial learning is an important contemporary research topic [22]. Some studies have proposed methods for creating effective classifier based intrusion detection systems [19, 9, 47]. Many studies have addressed the importance of data sanitization or adversarial influence at training time [8, 16, 34, 27]. Yet others focus on evasion of the deployed classifier [11, 30, 39]. We also focus on evasion of a classifier during operation, but instead of focusing on strategies for evasion, we propose a means of detecting these evasion attempts.

Many studies have addressed the topic of using diversity in ensemble classifiers to improve malware detection rates [26, 54, 33, 53]. Few studies, however, study practical strategies for detection of evasion attempts against these ensemble classifiers. Chinvale et al. proposed the use of mutual agreement between independent SPAM filters to optimize the re-train interval of the SPAM filters due to concept drift [14]. We use the same general approach. Our work differs in that we focus on detection of evasion on individual observations at test time. Our prediction is influenced by our mutual agreement analysis. Instead of multiple classifiers using independent data sets, we study the factors that make internal mutual agreement analysis effective in an ensemble using the same data sets across all classifiers in the ensemble. We also evaluate the effectiveness of our approach on direct evasion attacks.

The PDF classification problem has been studied extensively [17, 28, 31, 49]. Our empirical evaluation relies directly on PDFrate [45] which is the preferred choice for our study because it has been subjected to two separate recent evasion studies [30, 39].

3 PDFrate

PDFrate is a machine learning based malware detector operating on PDF documents. The pdfrate.com website allows user submissions and returns ratings for these submitted files. PDFrate is useful for this study because the underlying mechanisms are well documented [46, 45], it is openly available for online attacks, and it provides considerable information about each submitted PDF. Because of this transparency, PDFrate has been the target of practical adversarial learning studies. Having experienced independent evasion attempts makes PDFrate especially suitable for this study.

PDFrate classifies PDF documents based on analysis of their structural and metadata attributes. Risk factors for a malicious document include items such as existence of Javascript objects or improperly formatted timestamps. On the other hand, benign documents contain inert content such as text content or font objects. The basic structural and metadata information on which the features are based is extracted using regular expressions applied to the raw document. This small subset of structural information taken from the document is presented to the user in the document scan report. From this base information, features are extracted. Examples of features include the number of Javascript objects and the relative position of the end of file marker in the document. All told, 202 features are used.

Random Forests is used as the classifier in PDFrate.

Random Forests is an ensemble classification technique using individual trees, each of which votes and contributes to the final score. The number of tree (ntrees) is one of the primary tunable parameters and is set at 1000. The trees are constructed using a randomly selected subset (bagging) of the training data. The features used for splits at each node are also selected from a subset of the features dictated by the other primary parameter (mtry), which is set at 42. The original publications indicated that tuning these parameters had only minor affects on classification results and that Random Forests performed better than alternative machine learners including SVM. A discriminating characteristic of PDFrate is that it provides a score or rating instead of a simple benign/malicious determination. The score provided by PDFrate is the portion of trees that voted for the positive (malicious) class.

The PDFrate website also provides scores on three unique training sets. The Contagio data set is taken from a widely available dataset designated for researchers [35]. It contains 10,000 documents, evenly split between benign and malicious. The list of documents in this set is provided such that this training set can be replicated. The second data set was composed by researchers at a university and is called the University dataset. It contains a much larger number of documents, over 100,000, but the exact composition of the training set is not published. We use both of these training sets, the Contagio and University data sets, and the classifiers derived from them, in this study. There is another classifier, driven by community voting on submission, but this classifier has not been updated, presumably due to lack of community feedback. Beyond a benign/malicious score, some classifiers are built that seek to differentiate between targeted and opportunistic or commodity threats. We confine our study to the benign/malicious classification problem.

For this study, we obtained access to the source code of PDFrate and some PDF documents submitted to pdfrate.com in evasion studies. These items are generally not available to the public.

4 Mimicus

Mimicus [2] is a framework for performing mimicry attacks against PDFrate. It is the implementation of what is described by Šrndić and Laskov as “the first empirical security evaluation of a deployed learning-based system” [39]. As an independent, comprehensive, and openly available framework for attacks against the online implementation of PDFrate, it is well suited as an example of classifier evasion in our study.

Mimicus implements evasion attacks by modifying existing malicious documents to appear more like benign documents. Mimicus adds markers for additional structural and metadata items to documents. These additions do not involve adding actual content that is interpreted by a standards-conforming PDF reader, but rather these additions exploit a weakness in the feature extractor of PDFrate. The extraneous PDF attributes are added in slack, or unused space, immediately preceding the document trailer (structure at the end of the document), which is not prohibited by the PDF specification. Mimicus enables a very simple attack scenario. The attacker constructs a malicious document without concern for PDFrate evasion. Mimicus adds extraneous data that is skipped over by the reader being exploited but provides decoy structural elements that implement the mimicry attack against PDFrate. Differences between how malware is parsed by detection systems and the targeted program is a common problem [23]. This approach provides considerable flexibility in the evasion attack as the additional elements do not have to be valid. While some features can be decremented, this mimicry attack only adds fake elements to the document file–no existing elements are removed or modified.

Mimicus operates by comparing a malicious document to multiple different benign documents. The features are adjusted to make the malicious document mirror the feature vectors from the benign documents to the degree possible. These adjustments are bounded by the modification approach Mimicus uses. The candidate mimicry feature vectors are run through a local PDFrate replica to determine the scores. The best feature vector is selected. That feature vector is used as the goal in modifying the original malicious document by adding decoy structural and metadata elements. Due to interrelated features and other complications, it is not feasible to construct a final mimicry malicious document that exactly matches the target mimicry feature vector. The resulting mimicry malicious document has a feature vector that is somewhere between that of the original Trojan document and that of a benign document. After the mimicry document is created, it is submitted to pdfrate.com for evaluation.

An important observation of the Mimicus study is that the interdependency of PDFrate’s features make mimicry attacks more difficult because modifying one feature necessarily affects other features. It is generally accepted that irrelevant or redundant features are not desirable for machine learning methods. However, in the case of PDFrate, this attribute appears to make evasion attacks like those implemented by Mimicus more difficult by making construction of a PDF matching a target feature vector more difficult.

In order to perform this attack, a complete replica of PDFrate is required for offline analysis (the source code of PDFrate is not openly available). The most difficult component to imitate, and the component comprising the bulk of the code, is the feature extractor. This component uses regular expressions to extract features based on the structure and metadata of the document. 135 of the 202 features of PDFrate are publicly documented. Mimicus implements these 135 features, or approximately 2/3 of the features of PDFrate. Of these 135 features, Mimicus can influence 68, or approximately 1/3 of the features used by PDFrate.

One of the most remarkable findings of the Mimicus study is that the offline replica of PDFrate provides nearly identical results as those from pdfrate.com despite using only 2/3 of the pdfrate.com feature set and using a fully independent implementation. In the original PDFrate study, Smutz and Stavrou demonstrated that classification error decreases as the number of features used by PDFrate increases. With a randomly selected subset of the 202 features, classification error dropped rapidly until at least 20 features were used and continued to drop with diminishing improvements. Above 130 features, no improvement to classifier accuracy was observed. It is possible that the additional features add no discriminatory power because the existing features are already adequate for classification or the additional features are redundant. Having access to the PDFrate implementation, we were able to inspect these features unknown to Šrndić and Laskov. The majority of these features were based on relatively rare structural items Not surprisingly, these features were ranked low: none of these features was ranked in the top 20 and only 14 of these 67 features were ranked in the top 100 (out of 202) features. Due to experimental results in the PDFrate and Mimicus studies, as well as our own observations, we simplify our study by not performing any comparisons of results based on these two divergent feature sets (or feature extractor implementations). We consider them functionally equivalent as they provide extremely similar results.

All Mimicus attacks assume knowledge of the feature set used by PDFrate. There is no attempt to develop an independent or surrogate feature set as is the case with both the training data and the classifier used by PDFrate. It follows that for a mimicry attack to be successful, at least knowledge of the type of features is necessary. Also, since this attack leverages a difference between normal PDF readers and the PDFrate feature extractor, knowledge of how to exploit this difference is also necessary. Hence, all Mimicus attack scenarios are labeled with an "F" indicating that the attacker used knowledge of the features.

Relying on the common basis of the feature extraction, the Mimicus attacks demonstrate various levels of knowledge used by the attacker. In situations where the training data and classifier are known by the attacker, replicas that are very close to the original are used. In other situations, a reasonable replacement is used. The labels "T" and "C" are used to denote attacker knowledge of training data and classifier, respectively. Hence, an attack scenario with the label "FTC" denotes attacker knowledge of all three major facets of PDFrate.

The training set used by the Contagio classifier of PDFrate is publicly documented and is readily available to researchers. Hence, in attack scenarios where the training data is known by the attacker, the same data set is used by PDFrate and Mimicus. For scenarios where the attacker has no knowledge of the training set, Šrndić and Laskov compiled a surrogate training set with malicious documents sourced from VirusTotal and benign documents sourced from the Internet. In addition, they selected 100 malicious documents from within the Contagio training set for the baseline attack documents. We use the same Contagio and baseline attack datasets as those used in the Mimicus study.

Lastly, to complete the offline PDFrate replica, Šrndić and Laskov used a Random Forests classifier when knowledge of the classifier was known, and a Support Vector Machine classifier to simulate the case of the naive attacker. The Mimicus study shows that when all three particulars of PDFrate are spoofed, the result is nearly identical scores from the PDFrate online and the Mimicus offline classifier, despite various implementation differences.

Mimicus also implements a GD-KDE attack which seeks to attack the SVM surrogate classifier directly. This attack does not apply to Random Forests classifiers, and therefore does not directly apply to PDFrate. We discuss this attack further in Section 11.

5 Reverse Mimicry

Maiorca et al. also study evasion against PDFrate and other PDF document classifiers [30]. They style their attack a "Reverse Mimicry". Instead of adding content to a malicious document to make it appear benign (as Mimicus does), they embed malicious content into a benign PDF, taking care to modify as little as possible. The Reverse Mimicry attack is useful for our evaluation because it implements an independent evasion attack against PDFrate.

Three different evasion scenarios are implemented by Maiorca et al. In the EXEembed scenario, a malicious executable is implanted in an existing benign PDF document. The malware is executed when the document is opened. These documents utilize CVE-2010-1240. In the PDFembed scenario, a malicious PDF is embedded into a benign PDF. These embedded documents are rendered automatically when the document is opened. For evaluation, Maiorca et al. embedded a document implementing CVE-2009-4324 into existing benign PDF documents. Lastly, in the JSinject scenario, malicious Javascript, the same used in the PDFembed embedded document, is injected directly into the root benign document.

In order to evade detection, the Reverse Mimicry attacks focus on changing the document structure as little as possible. For example, in the EXEembed attack, a new logical version of the PDF is constructed with few new structural elements, but all the content from the original PDF is left in the file. A compliant reader will not display the content associated with the previous version of the document, but the artifacts will be analyzed by the feature extractor of PDFrate and similar detectors.

In addition to minimizing the structural artifacts of the malcode injection, Maiorca et al. make use of PDF encoding, especially stream compression, to hide the inserted content. For example, in the PDFembed attack, the malicious document is embedded in a compressed PDF stream. Detection tools, such as PDFrate, that do not decompress the PDF streams are not able to extract features from the embedded malicious PDF.

The Reverse Mimicry attack provides a strong complement to the evasion attempts of Mimicus. Mimicus uses addition of decoy objects that would not be processed by a normal PDF reader but are parsed by the simple regular expression based processing of PDFrate. The Reverse Mimicry attacks, on the other hand, use valid PDF constructs to minimize and hide malicious indicators. Mimicus operates by adding camouflage while the Reverse Mimicry attack seeks to make the malicious elements stealthy.

6 Mutual Agreement Analysis

Our goal is to show that an ensemble classifier provides information useful for determining the trustworthiness of the classifier on a given observation. We use the mutual agreement of individual classifier results to indicate when the ensemble classifier is suitable for use on a given sample. If the votes concord, then the result falls within the space adequately covered by the strength of the features and training data of the classifier. If the individual votes do not agree, then the resulting classification is suspect. The level of classifier mutual agreement is quantified in a score which is compared to a threshold for determining which observations are labeled uncertain.

An ensemble classifier operates by obtaining the result of many independent classifiers and combining the results to make a composite result. Typically, the result is combined by voting, where each independent classifier gets an equal vote. The votes are summed to generate a score. Traditionally, if the score is over 50% the observation is labeled malicious and otherwise the result is benign.

We seek to identify poor classification results by dividing the voting scores into sections where the individual classifiers agree or disagree. Hence, instead of splitting the vote result space in simple halves, we split it into 4 quadrants. In the 0 - 25% region, the majority of the votes agree that the result is negative (benign). Similarly, in the 75 to 100% region, the majority of the votes agree that the result is positive (malicious). However, if the score is between 25% and 75%, the individual classifiers disagree. To support comparison with simple ensemble voting predictions, this area can be split into the other two quadrants: Uncertain (Benign) from 25 - 50% and Uncertain (Malicious) from 50 - 75%. These classification outcomes are demonstrated in Table 1.

Table 1: Ensemble Classifier Outcomes

Voting Score	Outcome	
[0,25]	Benign	
(25,50)	Uncertain	(Benign)
[50,75)		(Malicious)
[75, 100]	Malicious	

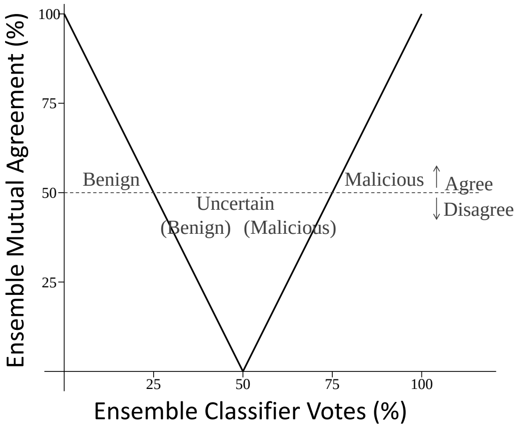

Figure 1: Mutual Agreement Based on Vote

To be more precise about this concept, we introduce a metric to quantify the agreement between individual votes in an ensemble classifier:

$$A = |v - .5| * 2$$

Where A is the ensemble classifier agreement rate and v is the portion of votes for either of the classes. This function is demonstrated in Figure 1, which also shows the classifier outcomes resulting from a 50% mutual agreement threshold. The end and middle points drive the general shape of this function. It follows that if the classifier vote ratio is either 0 or 1, then the classifier has full agreement on the result and the mutual agreement should be 1 (or 100%). If the classifier is split with .5 of the votes for each class, then the mutual agreement should be at the minimum of 0 (or 0%). As long as a single threshold is used, it is not important what shape is used for the lines between these end and middle points–any continuous curve would allow the selection of a given threshold on the classifier vote scores. The function need not follow the distribution of scores, for example. We choose a linear function because it is straightforward.

The threshold for agreement, naturally set at 50%, is the boundary above which the classifier is said to be in a state of ensemble agreement and the resulting classification should be considered valid. Below this mutual agreement rating, the classification is specious. This indicates that the classifier results are volatile due to factors such as inexpressive features or inadequate training data. We use the boundary of 50% throughout most of this paper. However, similar to the threshold of votes for classification which is usually and naturally set at 50%, this value can be adjusted by the operator. Decreasing this threshold decreases the number of observations in the disagreement or uncertain classification zone. Tuning of this threshold is discussed in detail in Section 10.

This mutual agreement rating can alternatively be described as the percentage of votes that remain after opposed votes are removed. For example, given a vote of 80% for the positive class, the 20% negative votes cancel out an additional 20% of the total votes. After 40% of the votes are removed, 60% of the votes remain. Hence, the mutual agreement rating is 60%.

This approach rectifies the perverse situation where a tie or near-tie in voting results in a classification that is generally considered the same as one where all votes are cast for the same class. For example, an outcome where 49% of the votes are cast for the positive class results in a mutual agreement score of 0.02 or 2%.

We have found that the mutual agreement metric is useful for evaluating the samples on which the classifier performs poorly. Moreover, we will show that the mutual agreement based prediction of uncertain is useful for elucidating those observations that are most likely to be classified incorrectly. We will further show the effectiveness of mutual agreement analysis at identifying observations that would evade detection due to both concept drift and direct evasion.

7 Evaluation on Virustotal Data

We measured the mutual agreement of PDFrate scores for Virustotal submissions during the year following the latest re-training of the University classifier (October 2013). From a corpus of PDF documents organized by initial upload to Virustotal, we randomly select 500 benign and 500 malicious documents per month. We consider any sample that has a detection by 3 or more AV engines as malicious and any that has less as benign.

Table 2: Counts of known Malicious documents from VirusTotal for each PDFrate prediction.

University Classifier

	Benign		Malicious	
Date		Uncertain		
201311	7	4	11	478
201312	2	0	2	496
201401	2	1	20	477
201402	10	6	16	468
201403	2	20	19	459
201404	9	10	19	462
201405	3	4	4	489
201406	20	9	22	449
201407	11	2	8	479
201408	20	18	22	440
201409	2	25	14	459
201410	7	21	5	467
total	95	120	162	5623

Contagio Classifier

total	841	1246	667	3246

Table 3: Counts of known Benign documents from VirusTotal for each PDFrate prediction.

University Classifier

	Benign		Malicious	
Date		Uncertain		
201311	479	19	0	2
201312	494	5	1	0
201401	483	14	3	0
201402	480	19	1	0
201403	493	6	1	0
201404	492	5	2	1
201405	490	9	0	1
201406	483	17	0	0
201407	485	14	0	1
201408	482	18	0	0
201409	491	9	0	0
201410	483	17	0	0
total	5835	152	8	5

Contagio Classifier

total	5638	280	72	10

Table 2 contains the two PDFrate classifier outcomes for the malicious samples, and Table 3 for the benign samples. We present monthly results for the University classifier, but for brevity, only present the year total for the Contagio classifier. These tables present the number of documents that receive classifier ratings of benign, uncertain, or malicious. We keep the convention of showing the split in the middle of the uncertain region based on a 50% score, allowing better comparison to standard classifier predictions and better showing the distribution of the scores. Generally, these tables demonstrate that the classifiers cast the majority of their votes for the correct class, malicious and benign respectively. The counts drop off rapidly through the uncertain outcomes and the incorrect class is a rare outcome. The distribution of ensemble classifier voting scores for the University classifier is shown for the malicious PDFs in Figure 2 and for the benign PDFs in Figure 3.

The primary observation is that using mutual agreement to add an additional outcome or prediction of uncertain dramatically decreases classifier error. This comes at the expense of a small number of observations receiving a prediction of uncertain. Table 4 compares

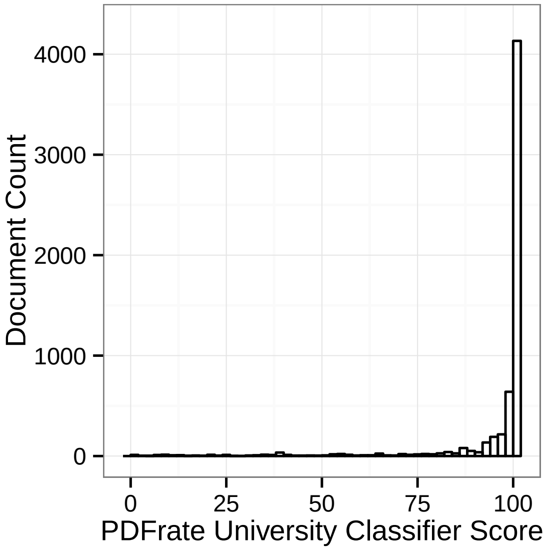

Figure 2: Scores for VirusTotal Malicious Documents.

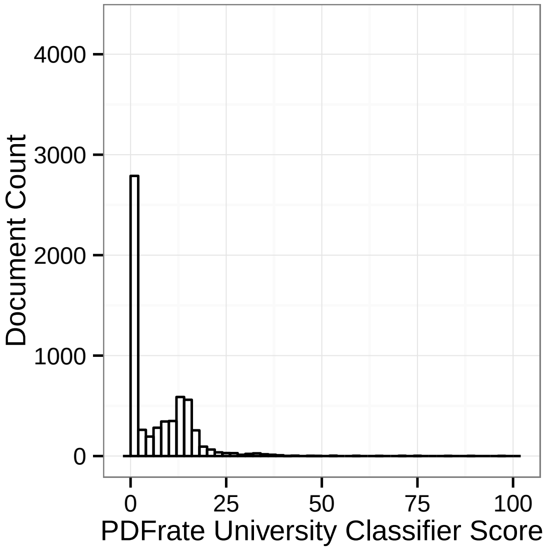

Figure 3: Scores for VirusTotal Benign Documents.

Table 4: Comparison of classifier performance using conventional vote threshold and mutual agreement derived Uncertain Rate (UR).

University Classifier

	FPR	FNR	UR
Conventional	0.22%	3.57%	-
Mutual Agreement	0.08%	1.58%	3.68%

Contagio Classifier

	FPR	FNR	UR
Conventional	1.37%	34.8%	-
Mutual Agreement	0.17%	14.0%	18.9%

the predictions of a traditional classifier with that using mutual agreement analysis. Classification error and uncertain rates are presented. For the University classifier, the false positive rate (FPR) drops from 0.22% to 0.08% and the false negative rate (FNR) drops from 3.57% to 1.58%. The trade-off is that 3.68% of the incoming observations are classified as uncertain. Of the observations labeled uncertain, 34% would be misclassifications with a traditional vote threshold. For the Contagio classifier, 54% of the uncertains would be classification errors. Note that we report the Uncertain Rate (UR) using the count of all observations as the denominator, while we use the conventional definition for FPR and FNR which use the count of the benign and malicious observations as the denominator.

One would typically expect the portion of classifier errors in the uncertain outcome to be under 50%, at least when counts of benign and malcious samples are equal. Even the most uncertain classifier, a random guess, should yield the correct prediction half the time. So even uncertain predictions should be correct about 50% of the time. Hence, 50% should be the normal upper bound for the classification error rate inside the uncertain outcome. The Contagio classifier exceeds this slightly because of it has an irregular score distribution.

If we are usually throwing away more correct classifications than incorrect, then what advantage does ensemble classifier internal agreement analysis provide? The uncertain range comprises a relatively small portion of the ensemble classifier scores but it captures a large portion of the misclassifications. Hence, by returning a result of uncertain for 3.7% of input, our classifier is able to increase accuracy from 98.1% to 99.2% on the remaining inputs. This additional measure of classifier confidence comes with no external validation of ground truth.

With the known classes of the samples labeled, it is clear that the University classifier is superior to the Contagio classifier. This is expected as the Contagio classifier contains over an order of magnitude fewer documents and was compiled nearly 3 years before the University classifier. Without any external knowledge, the mutual agreement analysis derived Uncertain Rates of 3.68% for the University classifier and 18.9% for the Contagio classifier gives us an objective measure of the relative confidence of these classifiers. These measures are very close to the ground truth misclassification rates of 1.9% and 18.1% respectively. The ability to estimate classifier error with no knowledge of ground truth makes the mutual analysis derived Uncertain Rate extremely valuable.

One surprising observation from this data is the lack of a steep decrease in classification accuracy throughout the year following the training of the classifier. It might be anticipated that the classifier would need to be retrained frequently to remain accuracy. We suspect that the low drift in the malicious documents over time was due to a lack of active evasion attempts against PDFrate. While polymorphism may be used to attempt to defeat signatures, rapid changes to the features used by PDFrate do not appear to occur in Virustotal submissions. We also tried to correlate exploits over time with classifier error and could discern no strong correlations between new software vulnerabilities and classifier evasion. In fact, the most common exploit found in the samples labeled uncertain was CVE-2010-0188, a very old, if not prolific, exploit. This exploit was the most common exploit reported in our VirusTotal submission data set. Our labeling of exploits was limited to the analysis provided by the cumulative detections of the AV engines in VirusTotal, which may introduce a bias in this analysis. To the degree our ability to correctly identify the exploits used in documents was not biased, it appears that new exploits are not associated with PDFrate evasion. It also appears that the various techniques used to defeat signature matching are generally orthogonal to the attributes that PDFrate uses for classification. This implies that PDFrate and signature matching techniques complement each other well.

We also analyzed the classification of individual trees in the Random Forest to see if some trees consistently performed better than others. Again, we could find no notable patterns: some trees were effective at some scenarios while performing poorly on others. In keeping with the stochastic nature of the generation of these trees, we could not glean any attributes correlated with evasion resistance in individual trees.

It is also noteworthy that the false negative rate is higher than the false positive rate and the contribution to the uncertain outcome is also higher from the malicious samples than the benign samples. This has a few implications. First, it seems that the classification PDFrate provides is more volatile for malicious samples than benign–possibly due to less variation in benign samples. We presented equal quantities of benign and malicious documents, but most environments are heavily skewed to benign observations. Hence, classification error and uncertain rates will drop in a typical, mostly benign, environment.

Despite covering half of the possible voting score range (using a 50% mutual agreement threshold), the uncertain result occurs relatively infrequently in practice because the bulk of the scores reside at the ends of the spectrum. Removing the observations with high ensemble classifier disagreement allows the classification error to drop dramatically. Mutual agreement analysis permits a higher degree of confidence in the outcome of a classifier without additional external information.

Table 5: Count of documents by PDFrate prediction for each Mimicus evasion attack.

	Benign		Malicious	
Scenario		Uncertain		
Baseline Attack	0	0	0	100
F_mimicry	2	70	26	2
FC_mimicry	7	78	15	0
FT_mimicry	10	64	26	0
FTC_mimicry	33	62	5	0
F_gdkde	7	92	1	0
FT_gdkde	4	95	0	1

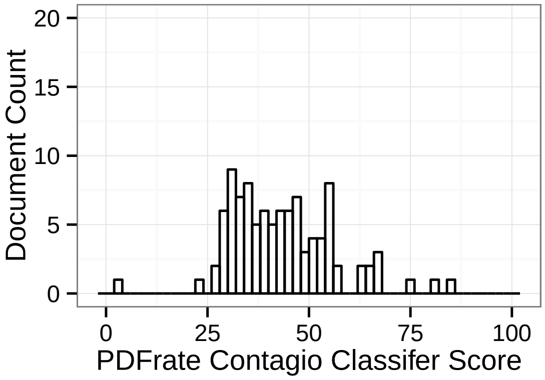

Figure 4: Score Distribution for F_Mimicry Attack.

8 Evaluation on Mimicus Attacks

To demonstrate the utility of mutual agreement analysis in identifying observations that evade detection, we reproduced the work of Šrndić and Laskov and applied mutual agreement analysis to these evasion attempts. We used the Mimicus framework to generate PDF documents that implement various evasion attack scenarios. We used the same data sets as the Šrndić and Laskov publication and submitted the resulting documents to pdfrate.com to obtain scores. Because we use the same attack data, our results are based on 100 samples per attack type. We were able to achieve results that closely mirrored those documented in the Mimicus study.

We present the results of classification using mutual agreement from the various attack scenarios in Table 5. Note that since all these documents are malicious, the correct classification is malicious. A rating of benign indicates successful evasion.

The distribution of PDFrate voting scores for the documents in each non-GD-KDE scenario is demonstrated in Figures 4 through 7. The GD-KDE attacks will be addressed specifically in Section 11.

In reproducing these results, some noteworthy observations are reaffirmed. First, knowledge of the target learning based detector's features (F), training data (T),

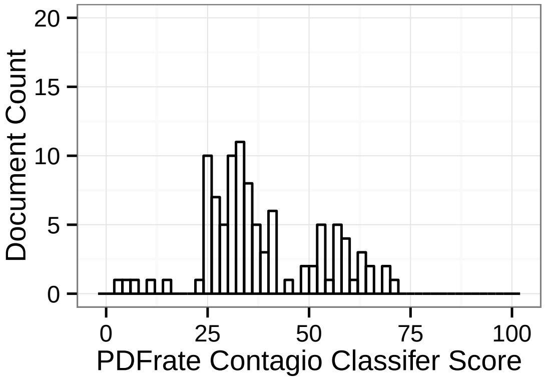

Figure 5: Score Distribution for FT_Mimicry Attack.

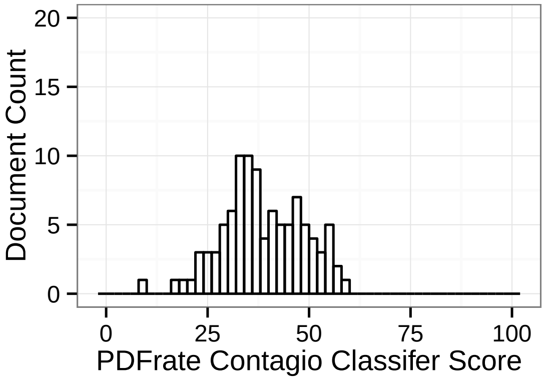

Figure 6: Score Distribution for FC_Mimicry Attack.

and classifier (C) are ranked. For an attacker, knowledge of the feature set is foundational and the most important when launching an evasion attack. Knowledge of the training set is next, and knowledge of the classifier follows closely. When combined, knowledge of all three components of the detector push many of the observations from the outcome of uncertain into a the domain of true evasion. Without knowledge of all three components, the vast majority of evasion attempts fall into the region of classifier disagreement.

Direct evasion attacks of this type are only possible with a good deal of knowledge of the classifier. If the details of the malware detector are kept secret, then evasion is much more difficult. However, if a detection system is to be deployed widely, attacker knowledge of the system is assumed. Of the three parts of a classifier based detector, it seems that the training data is the most likely to be customized on a per deployment basis. Many deployments will want to customize the training to match the samples seen in that environment. Also, the training data should be updated over time to account for natural drift. It is also feasible that the exact composition of a given deployment's training set could be kept secret from attackers. Indeed, pdfrate.com has multiple detectors using the same features and classifier parameters, but using different trainings sets.

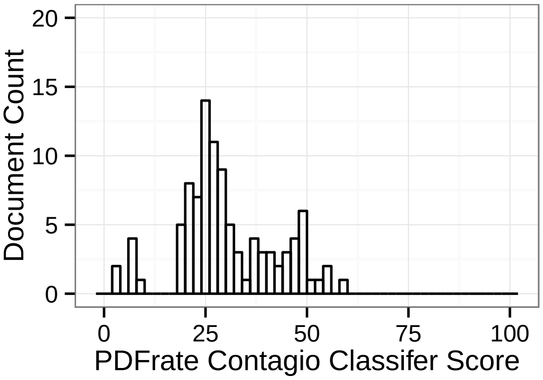

Figure 7: Score Distribution for FTC_Mimicry Attack.

In addition to evaluation against the Contagio dataset, the mimicry attack data was tested against classifiers trained with the University dataset. Throughout our study the University classifier was shown to be superior. Figure 8 shows the distribution of scores from applying the FTC attack scenario data to the University classifier. This results in what is an alternate FC attack scenario because the training set is unknown to the attacker. The results are very similar between the two classifiers. In both cases only 7 of the 100 evasion attempts are classified as benign. Carefully comparing Figure 6 and Figure 8 yields the observation that the University classifier provides a tighter cluster of scores near the center of the disagreement region. The results from the Contagio classifier are similar to that of the University classifier because the Mimicus evasion attempts use Contagio data for both baseline benign and attack data.

When at least one attribute of the detector is kept from the attacker, then most of the Mimicus evasion attempts fall within the disagreement region. This means that most of the evasion attempts are flagged as observations for which the classifier provides an uncertain result. Hence, if mutual agreement analysis is performed, then only a small fraction of the evasion attempts, 7% for the FC scenario, are succesful.

9 Reverse Mimicry Evaluation

We also applied mutual agreement analysis to the Reverse Mimicry attack proposed by Maiorca et al. Since the exact procedures required to reproduce these attacks are not known, we located the submissions made to pdfrate.com matching the description of these attacks for our evaluation. We are fairly confident that we were able to locate the Maiorca et al. submissions based on both the description of the attacks and analysis of server

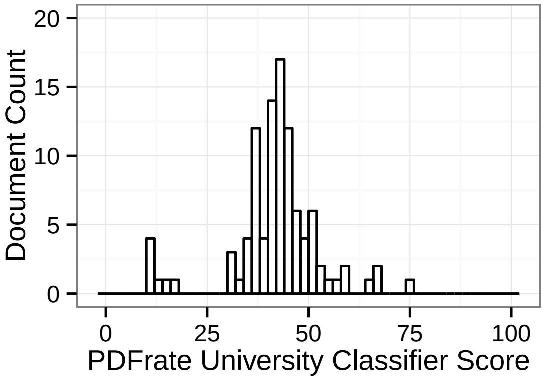

Figure 8: Score Distribution for FC_Mimicry Attack using University classier.

Table 6: Average PDFrate score from original Maiorca et al. publication and PDFrate submissions for each Evasion Attack using the Contagio classifier

	EXEembed	PDFembed	JSinject
Maiorca et al. Paper	7.1%	0.8%	14.8%
PDFrate Subset	17.0%	3.3%	31.4%
All PDFrate	17.0%	3.0%	31.6%

logs which showed an abnormally large number of PDF submissions from a single Internet address associated with their research institution.

Surprisingly, commodity AV signatures were quite useful in locating and separating these samples. The vast majority of the EXEembed samples are detected by Kasperky AV as "HEUR:Backdoor.Win32.Generic". The PDFembed and JSinject samples are both detected as "Exploit.JS.Pdfka.fbg" but are differentiated by the existence of the exact same PDF file embedded in the PDFembed samples. In all, we were able to locate about 400-500 unique samples of each attack type. To remain consistent with the Mimicus attack evaluation, we took a 100 sample random subset of each evasion attack for our evaluation.

Table 7: Count of documents by PDFrate prediction for each Reverse Mimicry evasion attack.

Contagio Classifier

	Benign		Malicious	
Scenario		Uncertain		
EXEembed	77	22	1	0
PDFembed	93	7	0	0
JSinject	30	67	3	0
total	200	96	4	0

University Classifier

	Benign		Malicious	
Scenario		Uncertain		
EXEembed	0	4	16	80
PDFembed	81	19	0	0
JSinject	0	22	55	23
total	81	45	71	103

While we were able to reproduce the qualitative results of Maiorca et al, we did have some variance in results. Table 6 shows our results compared to the Maiorca et al. publication for the Contagio classifier. We were not able to determine the exact cause of this variance. Nevertheless, the results are qualitatively very similar. PDFrate is effectively evaded if a traditional 50% score is used for the benign/malicious threshold. PDFrate is most effectively evaded by the PDFembed attack followed by the EXEembed and JSinject attacks. The University classifier was re-trained following the Maiorca et al. submissions. We used the newer version of the University classifier for our study so our results are not directly comparable with the University classifier results reported in their publication.

In Table 7 we present the results of applying mutual agreement analysis to the Reverse Mimicry attacks against both the Contagio and University classifiers. Here the University classifier demonstrates that it is much superior to the Contagio classifier. 67% of the Reverse Mimicry attacks are successful evasions (considered benign) against the Contagio classifier in spite of mutual agreement analysis. The University classifier fares much better, only being evaded by the PDFembed scenario.

The only complete evasions against the University classifier are achieved by the PDFembed attack. This attack is so successful because a complete malicious PDF is embedded in an otherwise benign document. This embedded document resides in a compressed data stream, which means that the structural features cannot be observed by PDFrate's feature extractor. This is in contrast to the other scenarios, EXEembed and JSinject, where despite efforts at minimization, some indicators of malfeasance are exposed.

To remedy the feature extractor evasion due to compression, these streams could be decompressed prior to analysis by PDFrate. However, this stream decompression is not useful for PDFrate style detections except in the case that the PDF document acts as a container for an embedded attack. Another, more direct approach to dealing with the embedded PDF document is simply to extract it and analyze it separately. Decompressing these objects is done by all PDF readers and some utilities allow these embedded objects to be exported. Indeed, many malware analysis and detection systems extract and perform analysis of PDF stream data [5, 1, 3]. This paradigm of embedded object extraction is not important solely for PDFs in PDFs, but for other file formats embedded inside PDFs or PDFs embedded in other file formats and containers. PDFrate will be equally ineffective at detecting a PDF embedded in an email, zip archive, or Word document if not extracted. We retrieved the embedded PDF using pdf-parser [50], but many PDF utilities can perform this operation. In all the PDFembed attacks, the embedded document was identical. The Contagio and University classifiers both easily detect this document with high confidence once it is extracted, returning scores of 97.6% and 100% respectively.

In the Reverse Mimicry evasion attempts, classifier internal mutual agreement analysis is able to label many of the otherwise false negatives as uncertain, preventing full evasion. However, in the case of the Contagio classifier, which is poorly suited to detecting these attacks, full evasion, or labeling a malicious sample as benign, is possible. This indicates that there is indeed a limit to which measuring mutual agreement can help an ineffective classifier.

When the stronger University classifier is used, mutual agreement flags most of the evasion attempts that would otherwise be successful. Furthermore, when the embedded PDF is addressed through extraction, and therefore detected, no evasion attempts are successful.

10 Mutual Agreement Threshold Tuning

For all of our evaluations, we used a 50% mutual agreement threshold, which splits the classifier voting score region into four equal sized quadrants. We also studied the impact of adjusting this threshold on our evaluation data. In Table 8 we present the University classifier outcomes applied to the VirusTotal and the FC Mimicus attacks.

The exact mutual agreement threshold chosen strikes a balance between improvement in classification error and the number of classifier predictions thrown out as uncertain. Operators who wish to have a lower amount of uncertain outcomes may choose a lower threshold. For example, if 30% is selected as a threshold, the uncertain region comprises ensemble classifier voting scores between 35 and 65% instead of 25 and 75% with a 50% threshold. The number of uncertain outcomes drops from 3.68% to 1.88% with the misclassification rates rising accordingly. The number of successful evasion attempts rises from 7% to 12%. The optimal setting for this threshold depends on the preferences of the operator. The sensitivity of uncertain detection is adjusted by tuning the mutual agreement threshold.

Table 8: University classifier performance as mutual agreement threshold is adjusted.

VirusTotal Data

Threshold	FPR	FNR	UR
0%	0.22%	3.57%	0.0%
10%	0.17%	3.32%	0.50%
20%	0.13%	3.03%	0.92%
30%	0.13%	2.13%	1.88%
40%	0.12%	1.78%	2.73%
50%	0.08%	1.58%	3.68%
60%	0.05%	1.25%	5.20%
70%	0.03%	1.03%	10.8%
80%	0.02%	0.75%	22.6%
90%	0.02%	0.32%	31.0%
100%	0.0%	0.05%	54.2%

Mimicus FC Attack

Mutual Agreement Threshold	Uncertain Score Region	FNR	UR
0%	-	84%	0%
10%	(45,55%)	69%	23%
20%	(40,60%)	31%	65%
30%	(35,65%)	12%	84%
40%	(30,70%)	7%	92%
50%	(25,75%)	7%	92%
60%	(20,80%)	7%	93%
70%	(15,85%)	6%	94%
80%	(10,90%)	0%	100%
90%	(5,95%)	0%	100%
100%	(0,100%)	0%	100%

11 SVM Counter-Evasion

In addition to demonstrating evasion against PDFrate, the Mimicus attack framework implements a Gradient Descent and Kernel Density Estimation attack against the classifier. This attack operates by exploiting the known decision boundary of a differentiable classifier [11].

We reproduced the GD-KDE evasion attacks of Mimicus and confirm that they are indeed extremely effective. Using the e1071 package of R [4] which relies on libSVM [13] we calculated the average SVM probability of 8.9% malicious (or 91.1% benign) for both GD-KDE scenarios, putting these attacks squarely within the evasion region. Šrndić and Laskov use the scaled distance from the SVM decision boundary to provide the same qualitative result. The GD-KDE attacks demonstrate that introspection of a single classifier such as SVM cannot be relied upon to detect evasions.

While effective against an SVM classifier, the results on PDFrate's RandomForest classifier using the GD-KDE attack are roughly comparable to the conventional coun-

Table 9: Number of documents per GD-KDE attack where Ensemble SVM classifier provides correct prediction as fraction of Features used in bagging is varied.

	Feature Subset			
Attack	5%	7.5%	10%	12.5%
Baseline Malicious	100	99	98	98
Baseline Benign	2	41	93	94
F_gdkde	100	100	99	5
FT_gdkde	99	100	92	1

Table 10: Number of documents per GD-KDE attack where Ensemble SVM classifier provides correct prediction as fraction of Training Data used in bagging is varied.

	Training Data Subset			
Attack	12.5%	25%	50%	100%
Baseline Malicious	86	87	92	98
Baseline Benign	100	100	100	100
F_gdkde	0	0	0	0
FT_gdkde	0	0	0	0

terparts (see Table 5). It is is not practical to wage a similar type of attack against RandomForests because the RandomForests has an extremely complex and stochastic decision boundary.

We sought to determine the extent to which we could make an SVM classifier more evasion resistant by enabling mutual agreement based uncertainty detection. We implemented a simple SVM based ensemble classifier using 100 independent SVM classifiers with the score being the simple sum of the votes of individual classifiers. To determine the attributes important to building subordinate classifiers useful for mutual agreement analysis, we varied the subset of features and training data used in constructing each of the individual SVMs. We performed a full grid search, but the most salient results are reported in Table 9 which shows feature bagging and Table 10 which shows bagging on training data. These tables demonstrate the portion of classifier outcomes that match the correct result (desired result for evasion attempts is malicious or uncertain). The application of random bagging to the many independent SVMs makes a GD-KDE style attack infeasible as there is no longer a single predictable decision boundary to attack.

It appears that bagging of training data is not particularly important in building an ensemble classifier where mutual agreement analysis is useful. To our amazement, we found no situation where anything but the full training set provided the best results. However, bagging of features is critical to constructing a classifier where mutual agreement analysis is able to identify uncertain predictions. It seems that the individual classifiers based on subsets of the complete feature set are much harder to evade collectively than a single classifier using all the features. While a single classifier can be evaded by successfully mimicking a subset of the features, it appears that a combination of multiple classifiers based on a small number of features requires a more complete mimicry across the full feature set.

Table 11: Count of documents by PDFrate prediction using optimal SVM ensemble classifier for each GD-KDE evasion attack.

	Benign		Malicious	
Attack		Uncertain		
Baseline Malicious	0	0	2	98
Baseline Benign	93	7	0	0
F_gdkde	3	97	0	0
FT_gdkde	8	91	1	0

The results also indicate that careful tuning of the portion of features used in bagging is critical. There seems to be a trade-off between the ability to correctly classify malicious observations (including evasion attempts) by using fewer features in each classifier, and benign observations by using more features. The use of fewer features results in a more complex classifier with smaller divisions while more features moves closer to standard SVM which has a single hyperplane divider. This result might be explained by suggesting that the features used in PDFrate provide better extrapolation for benign samples but that malicious samples have higher variation in PDFrate's features requiring more similar training samples for successful classification. This explanation is consistent with the results in Section 7 where PDFrate fared better on the benign samples over time.

Table 11 shows the outcomes of the optimal SVM ensemble classifier applied to the GD-KDE attacks and baseline benign and malicious samples. The result is that while the evasion attempts are successful in dropping the scores out of the malicious range, the vast majority of the evasion attempts fall in the uncertain range. Only 8% of the evasion attempts are fully successful in the best scenario while only 4.5% of the known data is in the uncertain region. These results are comparable to results obtained using PDFrate's Random Forest classifier where GD-KDE attacks are not possible.

12 Discussion and Future Work

Mutual agreement analysis provides a method to identify when a learning based detector is providing poor results, whether the cause is poor feature selection or inadequate training. Indeed, mutual agreement analysis can detect many evasion attempts using only classifier introspection. This ability to detect evasion is not absolute, however, and is limited by the quality of the base classifier. Mutual agreement analysis makes evasion more difficult, requiring mimicry across a greater portion of the feature set, but it is still possible.

Despite limitations, mutual agreement analysis was shown to make PDFrate able to detect most misclassifications in practice, whether caused by drift or direct evasion attempts. The addition of the uncertain outcome allows operators to know when their classifier is performing poorly, and to take action if desired. This should give operators confidence to deploy machine learning based detectors even when many aspects of the system are publicly known. As was shown in Section 8, local customization of the training data may be enough for an otherwise public detection system to be resilient against targeted evasion attempts. It might be advisable for operational systems to hide the exact scores returned from their classifiers as these scores assist attackers in knowing if changes they make hurt or help their evasion attempts. This information could weaken the benefit provided by a secret training set [22].

We can detect evasion, but we cannot fully prevent evasion, nor determine ground truth of uncertain observations. In most cases, a result of uncertain would require additional analysis and would result in either updates to the classifier or reliance on other detection methods. Since the uncertain results represent a small portion of the total observations, mutual agreement analysis may serve as a selector for expensive manual or automated analysis not practical on the larger data set.

Mutual agreement analysis is useful in evaluating machine learning detectors. A concise metric is the Uncertain Rate, or portion of observations for which a classifier is poorly suited to provide a prediction. The effectiveness of analysis using the mutual agreement score distribution and variance could be studied in the future. The classifier score distributions shown in Figure 2 and Figure 3 seem to indicate that regression could be used to predict the amount of successful evasions. The difficulty in this type of analysis, however, is separating the arcs for the benign and malicious data when external ground truth is not provided.

We found that feature bagging is the most important factor enabling mutual agreement based evasion detection. Our study also confirms previous findings that a large number of features may make evasion more difficult even if classification rates are not improved. Also, interrelated features may be helpful in preventing direct evasion by making construction of a malicious document that matches a target feature vector more difficult. Mutual agreement analysis should be applicable to any ensemble classifier. As we were able to convert SVM into an ensemble classifier supporting mutual agreement, so too should any classifier be able to be adapted to perform mutual agreement as long as feature bagging is possible.

13 Conclusions

We introduced a new technique to detect malware classifier performance degradation. To that end, we employ ensemble classifier mutual agreement analysis to evaluate the quality of classification rates by determining the samples on which the ensemble classifier prediction is unreliable. We applied our approach on 12,000 VirusTotal submissions, where we show that mutual agreement analysis improves the PDFrate false positive rate from .22% to .08% and the false negative rate from 3.57% to 1.58% by labeling 3.68% of the input samples as uncertain. In both the Mimicus and Reverse Mimicry evasion attacks, the majority of the evasion attacks are assigned the outcome of uncertain. While mimicry attacks are still possible, they must be much higher fidelity to be successful.

We believe that mutual agreement analysis can be applied generally. We show that even single classifiers can achieve mutual agreement based evasion detection through construction of an ensemble classifier using feature bagging. The Gradient Descent and Kernel Density Estimation employed with great success against Support Vector Machines is foiled by this approach. Ensemble classifier mutual agreement analysis provides a critical mechanism to evaluate the accuracy of machine learning based detectors without using external validation.

References

[1] jsunpack-n. http://code.google.com/p/jsunpack-n/.

[2] Mimicus. http://github.com/srndic/mimicus.

[3] pdfxray. http://github.com/9b/pdfxray_public.

[4] R project. http://www.r-project.org/.

[5] Ruminate IDS. http://ruminate-ids.org/.

[6] Suricata. http://suricata-ids.org/.

[7] VirusTotal - free online virus, malware and URL scanner. http://www.virustotal.com/.

[8] Barbara, D., Domeniconi, C., and Rogers, J. P. Detecting outliers using transduction and statistical testing. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, ACM, pp. 55–64.

[9] Barreno, M., Nelson, B., Sears, R., Joseph, A. D., and Tygar, J. D. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, ASIACCS '06, ACM, pp. 16–25.

[10] Bayer, U., Comparetti, P. M., Hlauschek, C., Kruegel, C., and Kirda, E. Scalable, behavior-based malware clustering. In *Network and Distributed System Security Symposium (NDSS) 2009*.

[11] Biggio, B., Corona, I., Maiorca, D., Nelson, B., Srndic, N., Laskov, P., Giacinto, G., and Roli, F. *Evasion Attacks against Machine Learning at Test Time.* No. 8190 in Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 387–402.

[12] Biggio, B., and Laskov, P. Poisoning attacks against support vector machines. In *In International Conference on Machine Learning (ICML) 2012*.

[13] Chang, C.-C., and Lin, C.-J. LIBSVM: A library for support vector machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

[14] Chinavle, D., Kolari, P., Oates, T., and Finin, T. Ensembles in adversarial classification for spam. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, ACM, pp. 2015–2018.

[15] Cova, M., Kruegel, C., and Vigna, G. Detection and analysis of drive-by-download attacks and malicious JavaScript code. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, ACM, pp. 281–290.

[16] Cretu, G., Stavrou, A., Locasto, M., Stolfo, S., and Keromytis, A. Casting out demons: Sanitizing training data for anomaly sensors. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pp. 81–95.

[17] Cross, J. S., and Munson, M. A. Deep PDF parsing to extract features for detecting embedded malware. http://prod.sandia.gov/techlib/access-control.cgi/2011/117982.pdf.

[18] Egele, M., Stringhini, G., Kruegel, C., and Vigna, G. COMPA: Detecting compromised accounts on social networks. In *NDSS, 2013*.

[19] Fogla, P., and Lee, W. Evading network anomaly detection systems: formal reasoning and practical techniques. In *Proceedings of the 13th ACM conference on Computer and communications security*, ACM, pp. 59–68.

[20] Handley, M., Paxson, V., and Kreibich, C. Network intrusion detection: Evasion, traffic normalization, and end-to-end protocol semantics. In *USENIX Security Symposium*, pp. 115–131.

[21] Heiderich, M., Niemietz, M., Schuster, F., Holz, T., and Schwenk, J. Scriptless attacks: Stealing the pie without touching the sill. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, ACM, pp. 760–771.

[22] Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B. I., and Tygar, J. D. Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, AISec '11, ACM, pp. 43–58.

[23] Jana, S., and Shmatikov, V. Abusing file processing in malware detectors for fun and profit. In *2012 IEEE Symposium on Security and Privacy (SP)*, pp. 80–94.

[24] Jang, J., Brumley, D., and Venkataraman, S. BitShred: feature hashing malware for scalable triage and semantic analysis. In *Proceedings of the 18th ACM conference on Computer and communications security*, CCS '11, ACM, pp. 309–320.

[25] Kakavelakis, G., Beverly, R., and Young, J. Auto-learning of {SMTP} {TCP} transport-layer features for spam and abusive message detection. In *LISA 2011, 25th Large Installation System Administration Conference*, USENIX Association.

[26] Kuncheva, L. I., and Whitaker, C. J. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. In *Machine Learning*, vol. 51, pp. 181–207.

[27] Laskov, P., and Lippmann, R. Machine learning in adversarial environments. In *Machine Learning*, vol. 81, pp. 115–119.

[28] Laskov, P., and Srndic, N. Static detection of malicious JavaScript-bearing PDF documents. In *Proceedings of the 27th Annual Computer Security Applications Conference*, ACSAC '11, ACM, pp. 373–382.

[29] Li, W.-J., Stolfo, S., Stavrou, A., Androulaki, E., and Keromytis, A. D. *A Study of Malcode-Bearing Documents*, vol. 4579. Springer Berlin Heidelberg, pp. 231–250.

[30] Maiorca, D., Corona, I., and Giacinto, G. Looking at the bag is not enough to find the bomb: an evasion of structural methods for malicious PDF files detection. In *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*, ASIA CCS '13, ACM, pp. 119–130.

[31] Maiorca, D., Giacinto, G., and Corona, I. A pattern recognition system for malicious PDF files detection. In *Proceedings of the 8th International Conference on Machine Learning and Data Mining in Pattern Recognition*, MLDM'12, Springer-Verlag, pp. 510–524.

[32] Mason, J., Small, S., Monrose, F., and MacManus, G. English shellcode. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, ACM, pp. 524–533.

[33] Menahem, E., Shabtai, A., Rokach, L., and Elovici, Y. Improving malware detection by applying multi-inducer ensemble. 1483–1494.

[34] Nelson, B., Barreno, M., Chi, F. J., Joseph, A. D., Rubinstein, B. I. P., Saini, U., Sutton, C., Tygar, J. D., and Xia, K. Exploiting machine learning to subvert your spam filter. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, USENIX Association, pp. 7:1–7:9.

[35] Parkour, M. 11,355+ malicious documents - archive for signature testing and research. http://contagiodump.blogspot.com/2010/08/malicious-documents-archive-for.html.

[36] Paxson, V. Bro: a system for detecting network intruders in real-time. 2435–2463.

[37] Perdisci, R., Dagon, D., Wenke Lee, Fogla, P., and Sharif, M. Misleading worm signature generators using deliberate noise injection. In *Security and Privacy, 2006 IEEE Symposium on*, pp. 15 pp.–31.

[38] Rajab, M. A., Ballard, L., Lutz, N., Mavrommatis, P., and Provos, N. CAMP: Content-agnostic malware protection. In *NDSS*, Citeseer.

[39] rndic, N., and Laskov, P. Practical evasion of a learning-based classifier: A case study. In *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, SP '14, IEEE Computer Society, pp. 197–211.

[40] Roesch, M. Snort - lightweight intrusion detection for networks. In *Proceedings of the 13th USENIX conference on System administration*, USENIX Association, pp. 229–238.

[41] Sahami, M., Dumais, S., Heckerman, D., and Horvitz, E. A bayesian approach to filtering junk e-mail. In *AAAI 98 Workshop on Text Categorization*.

[42] Schlumberger, J., Kruegel, C., and Vigna, G. Jarhead analysis and detection of malicious java applets. In *Proceedings of the 28th Annual Computer Security Applications Conference*, ACSAC '12, ACM, pp. 249–257.

[43] Shacham, H. The geometry of innocent flesh on the bone: Return-into-libc without function calls (on the x86). In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, CCS '07, ACM, pp. 552–561.

[44] Shafiq, M. Z., Khayam, S. A., and Farooq, M. Embedded malware detection using markov n-grams. In *Proceedings of the 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, DIMVA '08, Springer-Verlag, pp. 88–107.

[45] Smutz, C., and Stavrou, A. Malicious PDF detection using metadata and structural features. In *Proceedings of the 28th Annual Computer Security Applications Conference*, ACSAC '12, ACM, pp. 239–248.

[46] Smutz, C., and Stavrou, A. Malicious PDF detection using metadata and structural features. Available at http://cs.gmu.edu.

[47] Sommer, R., and Paxson, V. Outside the closed world: On using machine learning for network intrusion detection. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pp. 305 –316.

[48] Song, Y., Locasto, M. E., Stavrou, A., Keromytis, A. D., and Stolfo, S. J. On the infeasibility of modeling polymorphic shellcode. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, CCS '07, ACM, pp. 541–551.

[49] Srndic, N., and Laskov, P. Detection of malicious pdf files based on hierarchical document structure. In *Proceedings of the 20th Annual Network & Distributed System Security Symposium*, Citeseer.

[50] Stevens, D. PDF tools. http://blog.didierstevens.com/programs/pdf-tools/.

[51] Stringhini, G., Kruegel, C., and Vigna, G. Shady paths: Leveraging surfing crowds to detect malicious web pages. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, ACM, pp. 133–144.

[52] Varghese, G., Fingerhut, J. A., and Bonomi, F. Detecting evasion attacks at high speeds without reassembly. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, ACM, pp. 327–338.

[53] Waske, B., van der Linden, S., Benediktsson, J., Rabe, A., and Hostert, P. Sensitivity of support vector machines to random feature selection in classification of hyperspectral data. 2880–2889.

[54] Ye, Y., Chen, L., Wang, D., Li, T., Jiang, Q., and Zhao, M. SBMDS: an interpretable string based malware detection system using SVM ensemble with bagging. 283–293.